involve
a journal of mathematics

Algorithms for finding knight's tours on Aztec diamonds

Samantha Davies, Chenxiao Xue and Carl R. Yerger

msp

2017 vol. 10, no. 5

INVOLVE 10:5 (2017)
dx.doi.org/10.2140/involve.2017.10.721

Algorithms for finding knight's tours on Aztec diamonds

Samantha Davies, Chenxiao Xue and Carl R. Yerger

(Communicated by Ronald Gould)

A knight's tour is a sequence of knight's moves such that each square on the board is visited exactly once. An Aztec diamond is a square board of size $2n$ where triangular regions of side length $n-1$ have been removed from all four corners.

We show that the existence of knight's tours on Aztec diamonds cannot be proved inductively via smaller Aztec diamonds, and explain why a divide-and-conquer approach is also not promising. We then describe two algorithms that aim to efficiently find knight's tours on Aztec diamonds. The first is based on random walks, a straightforward but limited technique that yielded tours on Aztec diamonds for all $n \neq 22$ apart from $n = 17, 21$. The second is a path-conversion algorithm that finds a solution for all $n \leq 100$. We then apply the path-conversion algorithm to random graphs to test the robustness of our algorithm. Online supplements provide source code, output and more details about these algorithms.

1. Introduction

The problem of finding a knight's tour is one of many classes of chess-related problems that have been studied for hundreds of years. An early instance of such a tour was described by al-Adli ar-Rumi from Baghdad around the year 840 [Murray 1913]. Euler [1759; 1782] also studied the problem.

In chess, a *knight's move* on a board moves the piece horizontally by one square and vertically by two squares, or horizontally by two squares and vertically by one square. For clarity, Figure 1 indicates the possible moves of a knight. A *knight's tour* is defined to be a sequence of knight's moves on a board such that the sequence hits every square on the board exactly once. In an *open knight's tour*, there is no restriction on the starting and ending squares, whereas in a *closed knight's tour*, the starting square has to be one knight's move away from the ending square. We will focus mainly on closed knight's tours; if not stated otherwise, a knight's tour will

MSC2010: primary 05C45; secondary 05C57, 97A20.
Keywords: knight's tour, Aztec diamond, Hamiltonian, algorithm.

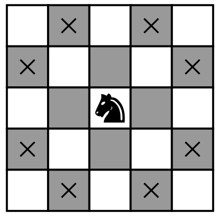

Figure 1. Possible knight's moves.

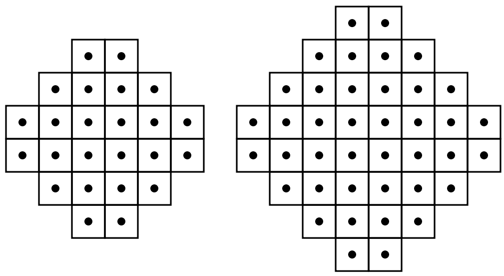

Figure 2. Aztec diamonds of radius 3 (left) and radius 4 (right).

refer to a closed knight's tour. We use the term *partial knight's tour* for a tour that visits squares only once, but not necessarily all of them.

Conditions for boards on which a knight's tour exists have been published for rectangular boards and variations thereof such as cylindrical chessboards [Watkins and Hoenigman 1997], toroidal chessboards [Watkins 2000], spherical (pillow) chessboards [Cairns 2002], and boards with deleted squares [Bi et al. 2015; Demaio and Hippchen 2009; Miller and Farnswort 2013]. Most proofs of the existence of knight's tours on these types of boards involve the expander method, made popular by Schwenk [1991]. With this method, one can take an open knight's tour on a board, add small strips of squares to extend the tour, and then use rotation, symmetries and induction to make a closed knight's tour on a new board.

One board that cannot be constructed by identifying edges of a regular chessboard is the Aztec diamond. We define an *Aztec diamond of radius n* as a lattice of squares in the $\mathbb{Z}^2$ coordinate system, whose centers (x, y) satisfy $|x|+|y| \leq n$ (these centers are composed of half-integral coordinates). Figure 2 shows Aztec diamonds of radii 3 and 4, with black dots representing the centers of those squares. These black dots are included because each square of the Aztec diamond chessboard corresponds to a vertex in the associated Aztec diamond graph. Two vertices in this associated graph are adjacent if their corresponding squares can be reached via a single knight's move.

In Section 2, we show that an Aztec diamond cannot be partitioned into smaller Aztec diamonds, which suggests that an inductive approach to finding knight's tours on an Aztec diamond is likely to fail. The symmetry of an Aztec diamond makes a divide-and-conquer algorithm appealing but we find that it is extremely difficult and sometimes impossible to divide an Aztec diamond. In Section 3, we introduce two algorithms for computing knight's tours (hamiltonian cycles): the *random-walk algorithm* and the *path-conversion algorithm*. The random-walk algorithm is deterministic, which means it will determine whether a knight's tour exists or not on a certain Aztec diamond upon completion. Our path-conversion algorithm is nondeterministic and hence cannot be used to disprove the existence of knight's tours on an Aztec diamond. But for Aztec diamonds, this algorithm is much more efficient than the random-walk algorithm in finding a knight's tour. In Section 4, we discuss some possible improvements on the path-conversion algorithm and several open problems in finding knight's tours on Aztec diamonds.

2. Theoretical results

Lemma 1. *The length of any closed knight's tour is even.*

Proof. First, we color the board with two colors, say black and white. Two squares are colored differently if they share a boundary. Hence, as shown via Figure 1, a knight can only move to a square that has a different color from its current square. In a closed knight's tour, the last visited square must be adjacent to the starting square, which means they are colored differently. Because the color alternates for every move, the number of squares visited in a closed knight's tour must be even. □

Lemma 2. *An Aztec diamond cannot be dissected into several smaller Aztec diamonds.*

Proof. We define the *border* of an Aztec diamond to be the set of squares that have a boundary that is not shared by two squares. By observation, the degree of a vertex on the border of any Aztec diamond can be 3, 4, 6 or 8. However, if we try to cover a top square of an Aztec diamond with a smaller one, we will obtain a square s, a degree 2 square, on the border of the remaining uncovered graph. Hence, this square cannot be a part of an Aztec diamond, which means that no Aztec diamond can be dissected into smaller Aztec diamonds. Figure 3 is a graphical representation of the arguments made in this proof. □

A *quadrant* of an Aztec diamond consists of squares whose centers are in the same quadrant of the coordinate system. For example, the first quadrant of an Aztec diamond of radius 4 is shown in Figure 4.

Lemma 3. *No knight's tours can be found in a quadrant of an Aztec diamond. In addition, an Aztec diamond of radius n cannot be dissected into four closed partial knight's tours of the same length, if $n \equiv 1, 2 \mod 4$.*

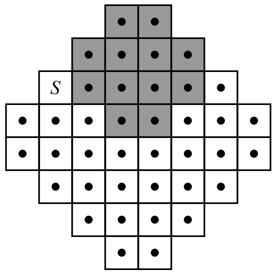

Figure 3. Visual proof of Lemma 2.

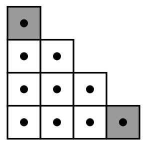

Figure 4. First quadrant of an Aztec diamond of radius 4.

Proof. The analogue of the two shaded squares in Figure 4 in a quadrant of an arbitrarily sized Aztec diamond have degree 1 and thus cannot be part of a closed knight's tour. Now suppose that $n = 4k+1$ or $n = 4k+2$ for some $k \in \mathbb{Z}$. The total number of squares in such an Aztec diamond is $n \cdot (n+1) \cdot 2$. If it can be dissected into four closed partial knight's tours of the same length, each tour is of length $\frac{1}{2}n \cdot (n+1)$, which can be expressed as $(4\mathrm{k} + 1) \cdot (2k+1)$ or $(2k+1) \cdot (4k+3)$, in contradiction with the fact that a closed knight's tour must be even in length by Lemma 1. This completes the proof. □

3. The random-walk algorithm and its variants

Since finding a knight's tour is essentially finding a hamiltonian cycle, we first introduce a brute-force algorithm for finding hamiltonian cycles called the random-walk algorithm. Again, these algorithms are run on the graphs associated with knight's tour moves on the Aztec diamond as described in Section 1. We present this algorithm on a nondeterministic machine because it is more succinct than the deterministic version. Whether a graph has a hamiltonian cycle or not can be determined by the following nondeterministic machine:

$N =$ On input $\langle G \rangle$, where G is a graph:

1. If G has only one vertex, accept. Else, proceed to Step 2.
2. Pick an arbitrary vertex v of G as the starting vertex. Mark v as visited.
3. Nondeterministically choose an unvisited vertex that is adjacent to the last marked vertex and mark it as visited. If none can be marked, reject.

4. If there exists an unvisited vertex, go to Step 2. Else, proceed to Step 5.
5. If the last marked vertex is adjacent to v, accept. Otherwise, reject.

To transform this algorithm into its deterministic form, we have to try every possibility when picking a vertex as described in Step 3, and backtrack if the chosen vertex fails to produce a hamiltonian cycle. In order to make this algorithm run faster, we add the following rules:

1. If at any point the starting vertex has no unvisited neighbors but the graph still has unvisited vertices, abort the current branch.
2. If a vertex v is adjacent to the last marked vertex and has only one unmarked neighbor, choose v to be the next marked vertex (note: there is a chance that v is the end of the tour, but this requires that v is adjacent to the starting vertex and that this situation has not been encountered in the previous steps). If there is more than one vertex that has this property, abort the current branch.

Although these improvements do not affect the asymptotic running time of this algorithm, they do expedite the process significantly in practice. Unfortunately, since the size of this problem grows exponentially as the radius increases, the running time of completing this improved algorithm on a large graph is astronomical. Our implementation of the random-walk algorithm runs for over one week on a Macbook Pro (2GHz Intel Core i7, 4GB memory, 1333MHz DDR3) without completion on an Aztec diamond of radius 5. However, we discovered that the choice of the starting vertex in Step 2 will tremendously affect the time used to find a knight's tour. For example, if we start with a certain vertex on an Aztec diamond of radius 4, the algorithm could run for more than ten hours without giving us a result, whereas with the right choice of starting vertex, we might obtain a cycle in ten seconds. This finding leads us to the next version of the random-walk algorithm.

To modify the existing algorithm, we simply run the algorithm with a given starting vertex on a separate thread. If no result is obtained within a certain amount of time, say ten seconds, we switch to the next starting vertex. The program halts if we find a cycle or if all the vertices have been chosen. An obvious flaw of this algorithm is that it is no longer deterministic because it does not exhaust all possible cycles.

When using this algorithm in practice, we are able to find knight's tours on Aztec diamonds of radius 22 or less except for those of radius 17 and 21, as shown in the table supplement. We could theoretically prolong the search time of each thread to increase the chance of obtaining a hamiltonian cycle, but given the size of our graphs, even an extension of five seconds per thread would lead to an increase of one hour in the total search time. We stop at radius 22 because the amount of time used to complete a search exceeds three hours.

4. The path-conversion algorithm

The idea of this algorithm springs from Parberry [1997]. In this paper, the author divides a large rectangular board into smaller rectangular boards (usually into four pieces), finds structured knight's tours on those smaller boards, and uses these special structures to connect all the disjoint partial tours together. The exact same technique would fail to generate knight's tours on the Aztec diamond because an Aztec diamond cannot be dissected into Aztec diamonds of smaller radii by Lemma 2. The attempt of cutting an Aztec diamond into four equally sized pieces does not seem to be feasible based on our proof when $n \equiv 1, 2 \bmod 4$ in Lemma 3.

Now we present our new algorithm, and details about how each step is accomplished will follow in the subsequent paragraphs. The algorithm does the following things:

1. Find an open knight's tour on an Aztec diamond.
2. Cut this open knight's tour into disjoint closed knight's tours.
3. Connect these closed knight's tours together.
4. If there exists a knight's tour that cannot be connected to the rest of the knight's tours, reject. Otherwise, accept.

Step 1: Find an open knight's tour on an Aztec diamond. In theory, finding an open knight's tour on an Aztec diamond is as hard as finding a closed one since both are NP-complete [West 2001]. In practice, the time spent finding an open knight's tour is significantly less than the time spent finding a closed one because there are more open knight's tours than closed knight's tours. We could still use the random-walk algorithm described earlier without the final checking step (Step 5) to find an open knight's tour. But for the sake of efficiency, we decide to use Warnsdorff's [1823] heuristic rule (as used in [Ganzfried 2004]) to speed up the process. As Ganzfried pointed out, Warnsdorff's rule does not hold true for every open knight's tour and it fails more regularly when the size of the graphs increases. But for the scope of this paper, Warnsdorff's rule (with some slight modifications) proves to be successful.

Warnsdorff's Rule [von Warnsdorff 1823]. In picking the next move, always pick an adjacent, unvisited square that has the least number of unvisited neighbors.

There are different rules about tie-breaking if two unvisited squares have the same amount of unvisited neighbors, but we simply use an ordering system of the squares to break ties. We number vertices from top to bottom, moving from left to right along each row (as shown in Figure 5) and pick the square with a smaller number if a tie appears.

In order to present the following steps visually, we assume that an open tour *ABCDEFGH* is obtained as shown in Figure 5.

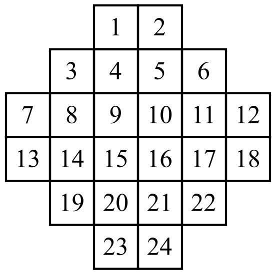

Figure 5. The ordering system to break ties in Warnsdorff's rule.

Step 2: Cut this open knight's tour into disjoint closed knight's tours. Suppose that we have obtained an open knight's tour in the previous step and that the tour starts at vertex v. Then label each vertex with a number indicating when it is visited. For example, v is the 1st vertex and a neighbor of v is the 2nd vertex (note that the labels in this step are different from the labels in Step 1, which are assigned based upon positions and used only to break ties). If a neighbor of v is the n-th vertex, then the vertices with numbers from 1 to n form a closed partial knight's tour because the piece can move back to v, so we add an edge between the starting vertex and the n-th vertex and delete the edge between the n-th vertex and the $(n+1)$-st vertex. Now we are left with a closed partial knight's tour consisting of vertices with numbers from 1 to n, and an open partial knight's tour consisting of vertices with numbers greater than n. We now let the $(n+1)$-st vertex be our new starting vertex and find another closed knight's tour using the above method. Repeat until every vertex in the original open knight's tour becomes a part of a closed partial knight's tour.

Since we eventually have to join these tours together, it is in our favor to make the number of closed knight's tours as small as possible. To achieve this goal, we use a greedy approach: we always pick the neighbor of the starting vertex with the greatest number to be the last vertex that closes the partial tour.

Figure 6 shows this procedure in action. On the left is an open tour; we first

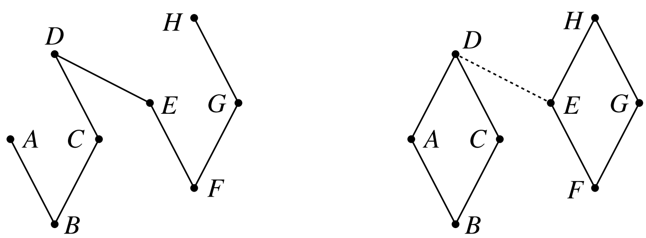

Figure 6. Left: an open tour on eight vertices. Right: the open tour cut into partial closed tours.

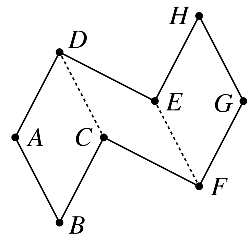

Figure 7. Joining partial closed tours into one closed tour.

pick A as the starting vertex and choose the 4th vertex D to close the partial tour. Then we pick the 5th vertex E as the next starting vertex and close the partial tour with H. Now every vertex is a part of a closed partial knight's tour and we move on to the next step.

Step 3: Connect these closed knight's tours together. Two knight's tours are able to connect to each other if a pair of adjacent squares in one tour is *parallel* to a pair of adjacent squares in the other tour. That is, the four squares in two parallel pairs must be able to form a closed partial knight's tour. To join these two tours, we delete the original edges between the four vertices and add the other pair of parallel edges. For example, *AB* is parallel to *CD*, and *DC* is parallel to *EF* in Figure 6, right. Hence, the partial tour *ABCD* can be joined to the partial tour *EFGH* as shown in Figure 7. Notice that our edge switching procedure is dependent on the fact that the knight's tour graph on the Aztec diamond has regular symmetrical structure, making easier switching opportunities.

In theory, the order in which we join these closed partial knight's tours matters because each time we join two tours, we change the direction of only two edges in these tours. In practice, however, we conjecture that it is less important overall and do not have a specific heuristic for joining tours. One ordering that may be helpful is joining the shorter knight's tours first before trying longer ones because the probability of whether a tour can be joined depends on the length of the tour. This ordering is hard to implement due to the data structure we use and thus ignored in our case.

Step 4: If there exists a knight's tour that cannot be connected to the rest of the knight's tours, reject. Otherwise, accept. This step checks whether all the closed knight's tours obtained in Step 2 are joined together. If so, then we have a complete knight's tour on the entire board. If not, we fail to construct a closed knight's tour with the given open knight's tour in Step 1. Because it is fast to find an open knight's tour with Warnsdorff's rule, we switch to a new open knight's tour if a closed knight's tour cannot be constructed from the previous one.

Figure 8 is an example of the entire algorithm on an Aztec diamond of radius 5. The left picture shows an open tour obtained in Step 1. The middle picture shows

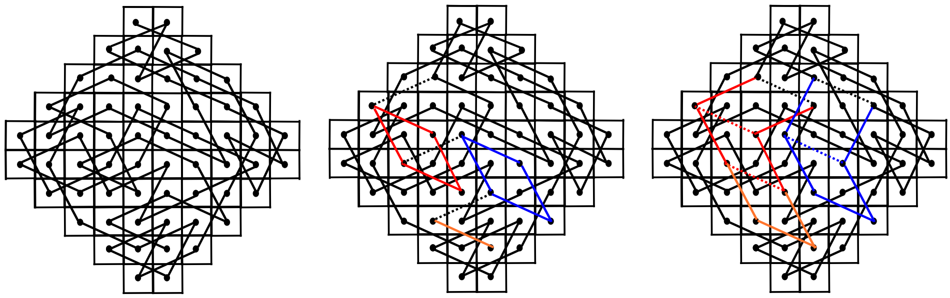

Figure 8. An example of the path-conversion algorithm on the radius 5 diamond.

how Step 2 of our algorithm cuts this open tour into four closed partial tours, each with a different color. The right picture shows how the Step 3 joins all the partial tours together.

With this new algorithm, we are able to find knight's tours on all Aztec diamonds of radius 100 or less as well as boards of radius 102, 104, 105, 106, 108, 109 and 111. The performance of this algorithm seems to deteriorate after the radius of an Aztec diamond exceeds 112 as we were only able to find an Aztec diamond of radius 125 via a search of boards having sizes between 112 and 140. This is probably due to the application of Warnsdorff's rule, which has a worse performance on large graphs. Our algorithm finishes in four minutes for a board of radius 100 or less and it grows polynomially as the radius increases because there is a one-to-one correspondence between starting vertices and open tours obtained in Step 1 of the algorithm (note that although the random walk algorithm can provide a knight's tour in 10 seconds, the algorithm takes a long time to finish even for an Aztec diamond of radius 5).

This new algorithm not only works for Aztec diamonds but also any graph because it is essentially an algorithm that transforms open knight's tours (hamiltonian paths) to closed knight's tours (hamiltonian cycles).

5. Applications to random graphs

We applied the path conversion algorithm to random graphs to test its robustness. Two questions were asked during this process:

(1) How many hamiltonian paths can be converted into hamiltonian cycles?

(2) Of all the graphs that have at least one hamiltonian path, how many have a hamiltonian cycle?

Note that the answers obtained below are not true answers but ones provided by our path conversion algorithm. Evidenced by the classic theorems of Dirac and Ore

[West 2001], sufficient conditions for the existence of hamiltonian cycles often involve degree constraints. By Ganzfried's claim [2004], Warnsdorff's rule fails more regularly when the size of the graph increases. Hence, we controlled for the average degree and the number of vertices when running our algorithm on random graphs.

5.1. ***Generating random graphs.*** Suppose we want to create a random graph with n vertices and (expected) average degree d. The total number of edges is $\frac{1}{2}dn$. Since a complete graph with n vertices has at most $\frac{1}{2}n(n-1)$ edges, we set the probability of existence of any edge between two vertices to $d/(n-1)$ so that the expected number of edges is $\frac{1}{2}dn$. Such a random graph, however, might not be connected. A disconnected graph has no hamiltonian cycles. Therefore, we ignore all disconnected graphs generated during this process.

5.2. ***Results on random graphs.*** To explore how average degree and number of vertices affect the answers to the two questions proposed at the beginning of this section, we conducted two sets of experiments. The first set fixed the number of vertices to be 1000 and changed the average degree, while the second fixed the average degree to be 8 and changed the number of vertices. The reason for choosing 1000 vertices for the first set is that it is a number small enough such that we could collect a decent amount of data in a short period of time and large enough such that a brute-force algorithm would take a long time to terminate. The reason for choosing average degree to be 8 is that it gives a rough comparison between the random graphs and the Aztec diamonds. Almost all vertices in an Aztec diamond have a degree of 8 except for those in peripheral areas.

The results are summarized in Table 1 and Table 2. The first column is the controlled variable, which could be either the average degree or the number of vertices; the second column measures, of all hamiltonian paths found using Warnsdorff's rule, how many can be converted into hamiltonian cycles; the last column shows how likely a graph contains a hamiltonian cycle if we know that at least one hamiltonian path can be found using Warnsdorff's rule. The statistics of each row is obtained from performing the path-conversion algorithm on exactly 1000 graphs. For convenience, we call the statistics in the second column the *conversion rate* and those in the third column the *success rate*.

From Table 1 we conclude that as the average degree of a graph goes up, it is more likely that a hamiltonian path can be converted into a hamiltonian cycle. In addition, more hamiltonian paths in total can be found. Therefore, the chance of finding a hamiltonian cycle rises significantly as the average degree increases. Similarly, if we fix the average degree and increase the number of vertices in a graph, we see a drop in the conversion rate. The number of found paths does not change monotonically. It rises first when the number of vertices changes from 100 to 200 and then falls when

degree	cycles/paths	cycles found/paths found
7	6.78%(4/59)	25.00%(1/4)
8	9.02%(1967/21804)	42.01%(92/219)
9	13.66%(19841/145207)	53.91%(338/627)
10	18.74%(75764/410166)	77.09%(700/908)
11	23.57%(141773/601410)	88.13%(861/977)
12	28.53%(212320/744178)	95.36%(946/992)
13	33.33%(272710/818283)	97.90%(979/1000)
14	37.96%(332630/876159)	99.30%(990/997)
15	42.40%(388014/915194)	99.90%(999/1000)

Table 1. Performance on random graphs with 1000 vertices.

vertices	cycles/paths	cycles found/paths found
100	69.67%(56676/81354)	87.73%(858/978)
200	50.47%(61280/121407)	76.29%(708/928)
300	37.44%(43992/117502)	64.90%(514/792)
400	28.34%(33034/116568)	60.03%(428/713)
500	21.87%(19129/87478)	48.58%(309/636)
600	18.27%(12085/66157)	44.18%(220/498)
700	14.14%(7323/51784)	46.55%(182/391)
800	12.42%(5068/40801)	44.03%(140/318)
900	10.35%(2868/27703)	37.96%(104/274)

Table 2. Performance on random graphs with expected average degree 8.

the number of vertices is further increased. The success rate, however, changes monotonically despite the oscillation in the number of paths found.

An Aztec diamond of radius 100 has 20200 vertices. According to these tables, the probability for finding a hamiltonian cycle on such a huge graph should be really small. Yet we were able to find hamiltonian cycles for all Aztec diamonds up to radius 100. Therefore, we conjecture that the degree distribution also affects how likely a graph has hamiltonian cycles. In the following subsections, we will test the performance of our algorithm on random regular graphs. But first, let us talk about how these graphs are generated.

5.3. ***Generating random regular graphs.*** To generate random regular graphs, we utilized the following algorithm described by Kim and Vu [2003]. Let n be the number of vertices in G and d be the degree of each vertex:

degree	cycles/paths	cycles found/paths found
7	2.67%(142/5328)	13.66%(134/981)
8	4.13%(1118/28549)	67.20%(672/1000)
9	5.94%(4300/80293)	97.60%(976/1000)
10	8.24%(11087/151781)	99.90%(999/1000)
11	11.07%(23181/241791)	100%(1000/1000)
12	14.28%(39929/323588)	100%(1000/1000)
13	17.91%(60671/401756)	100%(1000/1000)
14	21.86%(85501470826)	100%(1000/1000)
15	25.83%(114657/534245)	100%(1000/1000)

Table 3. Performance on random regular graphs with 1000 vertices.

1. Create a graph G with n vertices. Label these n vertices with integers from 0 to $n-1$. Create a list L with d copies of each integer.
2. Find two random integers i and j from L. While $i = j$ or vertex i and vertex j are already adjacent, choose another j. Connect vertex i and vertex j once i and j are chosen. Remove i and j from L.
3. If L is not empty, repeat Step 2. Else, output G.

Note that G may not be connected. Again we ignore all disconnected graphs. Furthermore, we can get stuck at Step 2 if we are unlucky. For example, L may contain integers of the same value. To avoid getting trapped in infinite loops, we restart our algorithm if no suitable pair of i and j can be found within a certain number of iterations.

5.4. ***Results on random regular graphs.*** To make comparisons more direct, we choose the same parameters. That is, all graphs generated during this experiment have exactly 1000 vertices. As shown in Table 3, the conversion rate on random regular graphs is much lower than that on random graphs. There are also fewer hamiltonian paths found on random regular graphs when the degree is larger than 8. However, random regular graphs have a better success rate than random graphs with the same parameters except for degree 7 (the statistics on random graphs of degree 7 are unreliable because the algorithm finds only 59 paths). One possible explanation is that the found paths are distributed more evenly on random regular graphs. Another possible explanation is that our algorithm works better for random regular graphs. Although the obtained results are algorithm-specific, it is worth asking whether degree distribution (instead of minimum degree) is related to the probability of finding a hamiltonian cycle.

6. Online supplement

Two supplementary files are provided online only. The code supplement contains Java programs implementing the algorithms described in the paper. The table supplement contains two Excel files: one shows all the results obtained from the revised random-walk algorithm on Aztec diamonds of radii from 2 to 20, and the other shows examples of knight's tours on Aztec diamonds of radii 30, 40, 50, 60, 70, and 80. Both folders include documentation.

7. Future work

Although our algorithm has worse performance on larger graphs, it can be improved in various ways. First, we can use better tie-breaking rules for Warnsdorff's rule or we can switch to another rule to find an open knight's tour. This will increase the likelihood of finding an open knight's tour, which is essential to the construction of a closed knight's tour. Second, we can have a different heuristic for cutting an open knight's tour into several closed partial knight's tours. A good cutting method will minimize the number of partial tours and possibly the variance in the lengths of these partial tours because it will reduce the probability of having a partial tour that is not able to attach to other tours. Third, it is likely that there exists a better order in which we join the closed partial tours.

Open Problem. Is there an Aztec diamond that has no open knight's tour?

Open Problem. Is there an Aztec diamond that has an open knight's tour but not a closed knight's tour?

References

[Bi et al. 2015] B. Bi, S. Butler, S. DeGraaf, and E. Doebel, "Knight's tours on boards with odd dimensions", *Involve* **8**:4 (2015), 615–627. MR Zbl

[Cairns 2002] G. Cairns, "Pillow chess", *Math. Mag.* **75**:3 (2002), 173–186. MR

[Demaio and Hippchen 2009] J. Demaio and T. Hippchen, "Closed knight's tours with minimal square removal for all rectangular boards", *Math. Mag.* **82**:3 (2009), 219–225. Zbl

[Euler 1759] L. Euler, "Solution d'une question curieuse qui ne paroit soumise à aucune analyse", *Mém. Acad. Roy. Sci. Belles Lett.* (*Berlin*) **15** (1759), 310–337. Reprinted in *Commentationes arithmeticae* **1** (1849), 337–355, and in *Commentationes algebraicae ad theoriam combinationum et probabilitatum pertinentes*, edited by L. G. Du Pasquier, Opera Omnia (1), **7** (1923), 26–56.

[Euler 1782] L. Euler, "Recherches sur un nouvelle espéce de quarrés magiques", *Verh. Zeeuwsch Genootsch. Wetensch. Vlissingen* **9** (1782), 85–239. Reprinted in *Commentationes arithmeticae* **2** (1849), 302–361, and in *Commentationes algebraicae ad theoriam combinationum et probabilitatum pertinentes*, edited by L. G. Du Pasquier, Opera Omnia (1), **7** (1923), 291–392.

[Ganzfried 2004] S. Ganzfried, "A new algorithm for knight's tours", REU proceeding, Oregon State University, 2004, available at https://www.cs.cmu.edu/~sganzfri/Knights_REU04.pdf.

[Kim and Vu 2003] J. H. Kim and V. H. Vu, "Generating random regular graphs", pp. 213–222 in *Proceedings of the thirty-fifth annual ACM symposium on theory of computing*, ACM, New York, 2003. MR Zbl

[Miller and Farnswort 2013] A. M. Miller and D. L. Farnswort, "Knight's tours on $3 \times n$ chessboards with a single square removed", *Open J. Discrete Math.* **3**:1 (2013), 56–59.

[Murray 1913] H. J. R. Murray, *A history of chess*, Oxford University Press, London, 1913.

[Parberry 1997] I. Parberry, "An efficient algorithm for the knight's tour problem", *Discrete Appl. Math.* **73**:3 (1997), 251–260. MR Zbl

[Schwenk 1991] A. J. Schwenk, "Which rectangular chessboards have a knight's tour?", *Math. Mag.* **64**:5 (1991), 325–332. MR Zbl

[von Warnsdorff 1823] H. C. von Warnsdorff, *Des Rösselsprunges einfachste und allgemeinste Lösung*, Varnhagen, Schmalkalden, 1823.

[Watkins 2000] J. J. Watkins, "Knight's tours on cylinders and other surfaces", *Congr. Numer.* **143** (2000), 117–127. MR Zbl

[Watkins and Hoenigman 1997] J. J. Watkins and R. L. Hoenigman, "Knight's tours on a torus", *Math. Mag.* **70**:3 (1997), 175–184. MR Zbl

[West 2001] D. B. West, *Introduction to graph theory*, 2nd ed., Prentice Hall, Upper Saddle River, NJ, 2001. Zbl

Received: 2015-07-28 Revised: 2016-07-08 Accepted: 2016-08-21

daviess@uw.edu *Padelfor Hall, University of Washington, W. Stevens Way NE, Seattle, WA 98105, United States*

cx94.main@gmail.com *Department of Mathematics and Computer Science, Davidson College, Box 7129, Davidson, NC 28035, United States*

cayerger@davidson.edu *Department of Mathematics and Computer Science, Davidson College, Box 7059, Davidson, NC 28035, United States*

involve

msp.org/involve

INVOLVE YOUR STUDENTS IN RESEARCH

Involve showcases and encourages high-quality mathematical research involving students from all academic levels. The editorial board consists of mathematical scientists committed to nurturing student participation in research. Bridging the gap between the extremes of purely undergraduate research journals and mainstream research journals, *Involve* provides a venue to mathematicians wishing to encourage the creative involvement of students.

See inside back cover or msp.org/involve for submission instructions. The subscription price for 2017 is US $175/year for the electronic version, and $235/year (+$35, if shipping outside the US) for print and electronic. Subscriptions, requests for back issues and changes of subscriber address should be sent to MSP.

Involve (ISSN 1944-4184 electronic, 1944-4176 printed) at Mathematical Sciences Publishers, 798 Evans Hall #3840, c/o University of California, Berkeley, CA 94720-3840, is published continuously online. Periodical rate postage paid at Berkeley, CA 94704, and additional mailing offices.

Involve peer review and production are managed by EditFlow® from Mathematical Sciences Publishers.

PUBLISHED BY

mathematical sciences publishers

nonprofit scientific publishing

http://msp.org/

involve

2017 vol. 10 no. 5

1944-4176(2017)10:5;1-8